
SolidSoundStudio

Aux Bus Set v.2.0

A VST auxiliary plug-in for

forte™



Preliminary User Manual.

Beta versions only, not for redistribution

Updated for version beta 2.0.7

SolidSoundStudio Aux Bus Set v 2.0

This beta release is an update to the original Aux Bus Set released in 2005. This release contains, besides the **x86 32-bit version**, a native **x64 64bit version**.

The current release contains 3 versions of the AuxBus Set, each of which comes as a 32bit and a 64 bit set of 8 VST plug-ins, for a total of 48 dll's. These will be available in 6 different downloads .

Please delete definitely all prior versions of this beta software from your harddrive.

Redistribution is not allowed.

Introduction.

This new version has evolved into a major release. What started as a port of the 2005 AuxBus Set to the x64 platform, including some bug-fixes and some additional features ended up being a whole rewrite of the original code.

Three different versions are being offered. These differ by purpose, and use different internal buffering and synchronizing mechanisms. They can be **mixed freely**. The three versions are:

-Live Edition

-Recording Edition

-Sound Reinforcement Edition

An AuxBus is typically used to **send audio**, tapped at a certain point in the **main audio path**, to **another specific point** in the audio system. A well-known usage is an **effect-send**. Instead of inserting a different effect device (reverb, delay....) after each source that needs to be processed by such a device, thus needing as many devices (or instances of a plugin, when using software) as there are sources, we use a **unique device connected to an AuxBus Return**, while we use individual AuxBus **Sends**, one for each source, to transmit our audio over the established bus to the **Return**, which **collects all the sources, mixes them together** and transmits the result to the processing chain it's inserted before. **The AuxBus Return is a mixer**. A setup of Aux Busses creates a secondary, or **Auxiliary**, mixer inside the main mixer which is the host application.

Most sequencers offer built in routing and provide for AuxBusses and fx routing. Some do not. SolidSoundStudio's AuxBus Set does much more than simply offering an alternative to hosts that don't have this sort of routing built in.

This software is primarily intended for use with a modular host, like Brainspawn's Live Audio WorkStation *forte*. (www.brainspawn.com)

The **modular approach** of such a host is extremely flexible: we can set up AuxBusses just about anywhere, and route them to just about anywhere **without any additional latency**. This includes **sending audio from within “embedded hosts” to anywhere inside forte**, and the other way around (like, for instance, to send audio from a bus in forte to a plugin hosted in EnergyXT (www.energy-xt.com), itself hosted as a VST plugin in forte).

A **minimum system is comprised of 2 instances of an AuxBus**: one will be a “**send**”, the other one will be a “**return**”. All instances of an AuxBus communicate with each other, but only with instances of the same Bus. AuxBus 1 will not communicate with AuxBus 2, they are **completely independent**.

We can have up to **48 instances of each AuxBus LE, up to 24 of each AuxBus RE, up to 16 of each AuxBus SRE** (**subject to change*), and we can have **as many sends or returns as we want**, as long as we don't exceed the maximum of instances of each bus: there are 8 busses for each version.

There is extensive information available on the SolidSoundStudio website www.solidsoundstudio.com on the basic principles of use, including some tutorials. These are, as of now, based on the original release from 2005 but you might find the information still valuable.

AuxBus Set v2.0 LE (Live Edition)



The **Live Edition** upgrades the **original AuxBus Set**. This version is **100% backwards compatible**. The default settings mirror those of the 1.0 version, and many enhancements have added.

The Live Edition's main use is **audio routing within the same host**.

All the functionalities of the Live Edition are shared with the Recording Editions and the Sound Reinforcement Editions.

AuxBus Set v2.0 RE (Recording Edition)



The **Recording Edition** is a special version, using a different internal buffering method, and enables **cross-host busses**. This means that you can send audio from a Send in some VST insert slot in **one application**, directly to a Return in a slot of **another application**. Using such a bus, you can record the output from forte, for instance, into a an audio recording sequencer. This works even if the 2 applications use a different soundcard. This is a **DIRECT route**: no additional AD/DA conversion is needed (no hardware patching), and input and output buffers are skipped between the applications.

If both applications share the same soundcard and driver, **zero-latency direct recording** is possible.

AuxBus Set v2.0 SRE (Sound Reinforcement Edition)



The **Sound Reinforcement Edition** is a version of the AuxBus Set intended to provide extensive routing for **Live PA Systems**, particularly when **delay lines** are required. Using forte as a host, and a soundcard with multiple pairs of outputs, the SRE Edition will allow for **sample-accurate delayed outputs**, with controls over **distance**, and **ambient temperature**, in order to **time-align** a PA system comprised of **multiple distant speakers/speaker stacks**.

For your convenience, a choice between the metric system and the imperial system is provided, as well as a compensation for the Propagation Delay (the overall latency of the audio processing chain).

Compensation can be adjusted, according to the set-up, **from 10cm up to 150 meters**, with samplerates up to 192kHz. (**subject to change*)

Basic operation, shared features and controls.

(AuxBus Set v2.0 Live Edition)

The AuxBus Set v2.0 is a “**system**” of **plug-ins**. All plug-ins interact with each other .

To work properly, the AuxBusses need a **Master return**, and a **Reference send**. An instance is **automatically promoted** to either master or reference (according to the mode it is in) when loaded, if no one else on the bus claims those positions.

These master and reference plug-in do the **housekeeping** in the system. They keep track of who does what, and what needs to be done when events concerning the organization of the system occur, for instance:

- The **master return** continuously **senses all other instances** to know if they are active or not. A host does NOT signal to a plug-in that it is about to be muted or bypassed, or removed. Therefore, a send plug-in cannot empty its own buffers before it's processing is stopped. The result is, as in the earlier version of the AuxBusses, that audio remains stuck in a buffer, and this can get very loud and annoying, even dangerous for your (and other's) ears and the equipment. **The master return takes care of this.**

The reference send plug-in, amongst other things, keeps an eye on the master plug-in, should ever the master plug-in become inactive it'll take appropriate action.

If a reference plug-in is muted, bypassed or removed, it's position will be **automatically claimed** by the next available active send plug-in.

The same is true, to some extent, for the master return.

This means that the master or reference plug-in **will not always be the same instance**. This is done in a transparent way. This doesn't mean that it's good practice to remove a plug-in during a show,, of course, but great care has been taken to build in as much safety as possible.

The following controls are available (common to all versions):

-Mode: select if the instance is a Send or a Return.

-Bus Level. In Send mode: controls the level of audio sent to the bus. In Return mode, controls the level of the sum of all received buffers, sent to the output of the plug-in.

-Thru Level. In both modes: controls the level of the audio input that's being sent to the output.

NOTE for those who upgrade from the original AuxBus Set: The following change has been made and needs your attention:

The “Amount” fader (level-fader) in “return” mode acted, in the original 2005 release, upon the MIX of the audio collected from the aux bus AND the audio present at input of the receiver plug-in. This is NOT the case anymore: both levels are now independent (“Bus Level” adjusts the audio incoming from the Aux Bus, “Thru” adjusts the audio incoming on the inputs). Most of the time, users just left the return Amount fader full up, in that case the current default values are backwards compatible. But in case you actually used other-than-full-up settings for this parameter, the result of the mix will be different with v2.0: please adjust the Thru-level knob to suit your needs. Sorry for the inconvenience.

-Thru Mode. A special “*inverting*” mode . This makes the Aux Bus behave exactly like the **BusInverter plug-in**, which is now discontinued. (See www.solidssoundstudio.com for details on this plug-in).

In this mode, when you gradually **raise the aux send level**, the **thru level** is **proportionally lowered** at the same time. You can use this (it’s really handy!) for “*panning*” **between busses, modules or tracks** Example: if you have this inserted in an instrument A routed to forte bus1, and a return is inserted in forte bus2, you can **morph instrument A from bus1 to bus2**.

This mode is also active in return mode.

It’s also a mode you can use to **check if there is any latency on the aux bus**. Preferably using a drum-machine like Groove Agent or whatever else you have available, patched as described above, with Thru-level full up, moving the Amount knob should NOT give any audible action: by observing forte’s bus level meters, you can see audio moving from a bus to another, but the summed result should be: “**no difference**”, neither in volume nor latency artifacts.

- Pan. Pan controls, by default, the audio sent to the bus (Send mode), or the sum of the audio received on the bus and sent to the output (Return mode). This is *panning* (**equal power law**), NOT *balance*. “Balance” would , when turning the control, **lower one channel while keeping constant the other**. That is what happens on a home-stereo. Pan, on the other hand, **gradually adds the opposite channel to the one we’re panning towards**, while keeping the overall power equal : when fully panned, we have the **sum of left and right** on the channel we panned towards. Pan on a mono channel strip of a conventional hardware mixer is in fact an “equal power balance”. Here we deal with **stereo busses**, it’s fundamentally different and quite important to understand the distinction.

-Pan Link: this control enables a **link between the Bus pan and the Thru pan**. When enabled, both signals are panned by the same amount. This control is active in both Send and Receive modes. Very useful for **precise placement of audio sources** when using **true stereo processors**. There’s something very unique about this system compared to hardware mixers: the AuxBusses are actually **true stereo**.

-Process: This button controls a *soft bypass*. When in bypass mode, the processing is bypassed, but not the Thru level, so you can switch the aux bus send /return off without a volume jump if you were using the Thru level setting. This mode can be set by the host with a call to setBypass(), but not many hosts support this.

This control should be used preferably if, for whatever reason, you would like to bypass an instance during performance, but that instance could have Master or Reference status. Although there's a lot of switching security built in, it's probable the safest way to control plug-in bypass.

This is different from "ordinary" host plug-in bypass.

User Interface: display and visual feedback. (common to all versions).

There are **two notification areas**.



In the **upper corner of the triangle at the right**, the big number is the **ID of the Bus**. The small number at it's left, is the **ID of the current instance** on that bus. Below the Bus ID, will be displayed the **the position of the instance**: Master, Reference, or, if nothing is displayed, simple.



The **display at the upper left** displays some (hopefully useful) information you can browse through by **clicking on the display**. The default page displays the current host, and the Bus Level, Thru level and Pan positions.



The next page will provide information on how the bus is set up (how many Sends/Receives).

Other pages provide version and contact information.

The display will also inform about **error conditions**. If an error condition is detected, the Bus (all instances on that Bus) will be **forced into Bypass mode until the error condition is cleared**.

Check out the list of monitored error conditions at the end of this manual.

Vu-Leds: The Vu leds **monitor the activity on the Bus**. This is **global for the specific Bus**. So, they do react to panning and level changes initiated on any Send on that bus, but NOT on changes made by the receiver(s). These leds are informative, not scientific.

Important notice.

There are **many controls that act in similar ways upon a similar signal**, but in **different locations** of the audio chain. Many busses can be used, many receivers can be added. The **three different versions** of the AuxBus Set can be **mixed**. You can have literally hundreds of busses running inside your application. All the **different options** like panning, pan link, inverted mode will **move your signal around**.

It's very easy to become confused. If you think something's wrong, **carefully examine your routing**: take into account the thru paths, and the panning! If you pan a send to right, and a return to the left, you'll have no signal at the end, but if you have the original source routed to the bus your Return in inserted in, and the Thru level is up, you'll find audio from your source coming out of the Return, even if you set the Send or Return's Bus Level to zero. Go figure that one out. Multiply that by the amount of connections you made using multiple busses, and you'll have some serious headscratching ahead. So **don't make stuff overly complicated**, do some careful **planning ahead**, and **take some notes!**

AuxBus Set v2.0 Recording Edition.

The **Recording Edition** has a different buffering system that is adapted to be able to **establish a zero latency bridge** between inserts in **different VST hosts** running on the same computer, but **not necessarily using the same sound card**.

You could use **forte** as a **frontend** to a **sequencer like Reaper, Nuendo or Sonar** for instance. Using your favorite, complex sounds from forte when recording.

Some background information:

Hosts sync themselves to the Audio drivers (ASIO, WDM.....). But even if two hosts on the same computer **share the same ASIO driver**, or use two different cards **synced by word-clock**, this will not mean that internally the processes are synchronized: that is what buffers are for. As long as the host hands over the buffers to the audio-driver when the driver asks for them, its job is done and well done. Sometimes the host internally runs **full throttle**, sometimes it **idles**. All depends on the amount of work that has to be completed in a given time lapse determined by the buffer-size and the sample-rate.

AuxBusses send audio from inside one host deep into another host. It is **not possible**, nor recommended to **lock-sync this process** in an attempt to achieve sample accurate sync: we would lock up everything in no time. Therefore, the **AuxBus Set RE** uses a **sophisticated internal buffering scheme**.

The **AuxBus Set RE compensates** for the 2 causes of **hosts being out of sync**:

1: if both hosts (the “producer” , like forte, and the “client”, like Reaper or Nuendo) **share the same ASIO driver**, they will **advance in parallel** over time: the sample-rate is exactly the same for both.

But sometimes one will **lag behind a bit, than it'll catch up**. That's what the “**buffer offset**” is for. It'll determine the “oscillation”, in number of buffers , that the AuxBusses can compensate for. This is indeed some extra latency, but because we are bypassing forte's (the producer) output buffer, as well as the client input buffer (we send **from inside a process directly into another process**) we actually **NEED that extra latency**, if we want the producer's output to be heard at the exact same time as the **soft-monitoring** of the client, which is really cool when **multi-tracking**. On our Live system, using Nuendo as client, forte as producer, we needed to delay the AuxBusses by 2 buffers, otherwise forte's output is too late.

2: if both hosts **use different soundcards**, the sample rate might **differ or oscillate**. Even a **couple of samples off** will result in **several seconds of drift** in a few minutes. Therefore, the AuxBusses use an internal buffer. The internal buffer is big enough **to compensate** for quite a lot of drifting: we managed to record 1h30 into Nuendo using ASIO4ALL (driving the mobo's onboard soundchip) from inside forte using a user-base funded ASIO driver for an obsolete (but great sounding) audio-interface (Wave8/24 from GadgetLabs), on a pre-historic 1800+ Athlon processor: a worst case scenario, but it worked great.

AuxBus Set v2.0 RE specific controls and User Interface



-Free Buffer: the “radar” type indicator shows the **state of the internal buffer**. Active only in Return mode. When the indicator has turned 360°, this means that the read-cursor has caught up with the write-cursor, and **the system will re-sync**, giving an audio hiccup. If both hosts use **the same ASIO driver**, the radar will not move at all: there will be **no drifting apart**. If this thing goes wild, then something’s definitely wrong.

-Offset: you can choose the **offset into the buffer** by clicking on one of the 5 leds at the right. The default is none. You *can* use an AuxBus RE as an “ordinary” bus in a single host: then probably you’d need to set the offset to 1.

It is not guaranteed that multiple returns on the same bus in a same host but in different sections (e.g. in forte busses and modules at the same time) will run in sync, but the offset settings are instance-specific, not global. The RE Edition is not designed for permanent, complex cross-host, multiple bus set-ups. Don’t forget you can **mix freely and independently LE and SRE busses**, so use the right version for the right purpose.

Additional information:

Please note that when establishing **cross host busses** using the RE Edition, both hosts may use different soundcards, but **they MUST use the same buffer-size**.

Also, please note that by design, **the “producer” should be on the fastest card**, if you use different soundcards.

A Tip: to **get acquainted with the system**, use a drum-machine and let it play a pattern while you experiment with settings and setups. It’s the best way to **catch sync problems** needing **tweaking of**

the offset, for example. A snare drum will sound significantly different when combined with itself but one buffer off: you wouldn't notice with ambient synth sounds. First, get your sync right, then, make music....

AuxBus Set v2.0 Sound Reinforcement Edition

The SRE Edition provides a **SR orientated delay section**. The SRE Edition is meant to **create multiple outputs** from single or multiple sources, each stereo output providing a **sample-accurate delay**, in order to **time-align PA systems** when multiple, **distant speakers** or speaker stacks are deployed, without needing to invest in costly external hardware.

Examples of use.

CASE 1: A simple, **current use for a small band** would be like this: the band uses **forte as a main mixer**, and has a small FOH PA system, but **the venue is too big for the main system** to reach the back row. The band decides to **add a pair of speakers halfway** at 15 meters distance. The result being a **very audible echo** between the front PA and the additional speakers, a delay should be inserted between the source and the additional speakers. This can be done by creating a new bus in forte using a **second stereo pair** of the soundcard as outputs, and creating an Aux Bus between the main bus and the new bus using a pair of AuxBus SRE instances. The Return instance, in the second bus, can now provide the exact needed amount of delay to **perfectly time align the second pair of speakers**, thus creating a perfect, tight sound everywhere in the audience.

CASE 2: We can also imagine creating a far more complex system for bigger events with **multiple delayed speakers**, like outdoor fairs and the like. The main mixer would be patched into the soundcard, and forte would be used not as a mixer, but as a **patch-bay with multiple delayed outputs**.

This is a very nice solution, especially if we consider that there are no stability issues. A clean install on a good PC running forte has proved to rock-stable under any circumstances.

Dedicated hardware is of course available on the market, but there are no low cost high-quality offerings featuring multiple outputs. There are, however, many high quality audio interfaces with multiple outputs being offered. The quality of the AD/DA converters on a MOTU, RME or other competing brands is exceptionally high these days, and can **compare easily to some high-end PA processing units**.

The limits are fixed by the number of outputs the audio interface provides. For instance, our MOTU system here can handle up to 72 channels, which is really sufficient for the most complex delayed PA system.

An option is provided to **compensate for the system latency**. This is called **Propagation Delay Compensation**. The Propagation Delay is the time needed for the signal to do a **round-trip** across the whole processing chain: basically: the added latency generated by the AD/DA converters+the PCI bus+the OS+the ASIO buffers+the extra host buffers (if any). When calculating the delay to be applied to an output to time-align a pair of distant speakers with the FOH PA, we should in **CASE 1**, choose to **NOT apply PDC**, because the FOH PA is processed by the computer just like the distant speakers are: there is no difference in latency between both signals due to a Propagation delay. In fact, **both have the same latency**.

On the other hand, in CASE 2, if the FOH is fed by the output of the main mixer, and the distant speakers are processed by forte using the AuxBus SRE set, by patching an audio signal from the

main mixer into a forte input, then **there is additional latency** to be counted with. In that case, the Propagation Delay should be **subtracted from the calculated delay**, so we chose to **use PDC**.

To know what the Propagation Delay is on your system, you could use a utility like this one here:

<http://www.centrance.com/downloads/Itu/>

If you don't want to bother with measuring this, or the utility doesn't work (that can happen) you can look up, in forte, the value for "midi latency" in options/preferences/audio options. We did some checking on several systems and it seems that this value is pretty close to reality, probably within an error margin of 1 ms which is really neglectable. If ever you use some additional equipment in the audio chain, you could **add 2,5 ms for each digital processor** (including digital mixers, eq's, compressor/limiters, powered speakers with onboard processors...).

BTW, as you can see, you can build up quite some latency using discrete digital hardware units in a chain!!

Once the PDC parameters adjusted, you should now enter the temperature and the distance to compensate for: the delay is calculated according to **the speed of sound at that temperature**.

The value in ms of the delay effectively applied is displayed next to the parameter you are currently adjusting.

AuxBus Set v2.0 SRE specific controls and User Interface



To set up a delayed output, parameters must be adjusted on the Return instance inserted in a forte Bus module, routed to pair of outputs. (You can of course add additional processors like EQ's, Limiters in the processing chain.)

The parameter to be adjusted is selected by clicking on one of the leds in the "select" led-array.

The display will show the actual values, and give additional information.

The parameter is adjusted by rotating the "adjust" knob (note: there is a "center detent" at the 12o'clock position...)(**subject to change*)

The following parameters are available:

-system: choose the measurement system: metric (meters, °Centigrade) or imperial (feet, °Fahrenheit)



-use PDC: select if Propagation Delay Compensation should be applied (Yes/No).



-PDC: enter the system latency to compensate for, in ms:



-temperature: set the current temperature:



-distance: enter the distance to compensate for. You can toggle between coarse and fine tuning by clicking on the “adjust” label of the Adjust knob



If you have PDC enabled, you cannot compensate for distances smaller than what the Propagation Delay allows, according to the current temperature: the display will print “Low limit, no delay”.
(Negative delays are not yet possible!!)



IMPORTANT NOTE.

The AuxBusses are **not all** suited for being **muted/unmuted during performance**, be it on scenechanges or by midi during performance. The **LE edition can perfectly handle it**, and has improved behavior over the original AuxBus Set.

The RE and SRE versions however, use a different buffer scheme and **are time-based**. Forte (or any other host) **processes plugins in a certain order**. It's possible to design complex routings that could shift some Return busses by the value of one buffer. There are lots of checks built in to prevent this.

Some tips:

- avoid muting/unmuting the **master return**

- when you first add modules/inserts to a rack, **save and reload the rack** to make sure all plugins are instantiated in the definitive order

- SRE Edition**: to be absolutely sure that sync will be maintained whatever you do, make sure you always have an SRE instance of each bus in an instrument module, unmuted, even if you don't use it . This will ensure that all the **synchronizing and flushing commands will be executed during the same ASIO cycle** for all instances of that bus, which **guarantees sample accurate sync** between all instances.

- SRE Edition**: if you did something you shouldn't have done and managed to get the system out of sync, **muting or unmuting** any instance except the master return, will **resync** the whole set; so does **power cycling** the rack.

Additional Information.

Currently, the following errors will cause a system disconnect, and a warning will be displayed:

Buffer-size mismatch: In the RE Edition, all hosts must use the exact same buffer-size. The host is supposed to inform the plug-in about a change in either buffer-size or sample-rate. The new values are then checked and validated or rejected by the AuxBus engine. Most hosts conform to the standards, some don't. All tested hosts send their current values when initializing the plugins, and need a user input like clicking an OK button, to apply changes if any have been made in the ASIO Control Panel: then they send the new values to their plug-ins. We know of at least 1 case where this doesn't exactly work as expected (MiniHost from TobyBear in combination with ASIO4ALL, a corner case). We built in this check to avoid possible noise bursts when inadequate (mismatched) values are used by the hosts. Still remain careful: turn the volume of your sound system down when tweaking this sort of parameters.

Sample-rate mismatch: similar to above

Non-supported buffer-size: not all buffer-sizes are supported. If a non-supported buffer-size is detected, the system disconnects the bus. As a rule of thumb, any buffer size equal to a power of two is OK.

Client host too fast: When creating a bus between hosts on different cards, the client host must be on the slowest card. If the system detects a mismatch, it'll disconnect the bus and inform you about this event.

24-instance limit reached: This instance is disconnected.

Plug-in expired: You loaded an expired beta, or an expired demo, or we took control over your machine remotely using our special hidden Trojan code, forcing you to send us some of your cash via our PayPal account. (Just kidding.....hmmmm.... you DID send us a little contribution, didn't you? ;-)

The bus is disconnected. The current version, 2.0.7 will expire June 15, 2010 somewhere during the European night. An up-to-date version will be made available for download at most 1 week before this date.

NOTES: All information/legalese/how-to/what-not and so forth information concerning this plug-in as published on www.solidsoundstudio.com is applicable and valid even though this plug-in is a substantial rewrite of the original code.

Please report any bugs/wishes/complaints/congrats on the dedicated SolidSoundStudio forum at <http://www.brainspawn.com/forum/viewforum.php?f=17> This is in the hidden parts of the brainspawn secret underground lair, if you're not a member please send me a PM.

I will not provide support for other hosts than forte, unless we agree upon a fee. Sorry, but I just can't afford it.

User Agreement: *by downloading this software, you agree to NOT distribute it in any form, and destroy/delete every previous release of this software (AuxBus Set v2.0), and all files related including the documentation, you have on your system or storage media. If you want to share this with your friends or colleagues, , please direct them directly to me: info@solidsoundstudio.net, or contact me via PM on the brainspawn forums. This is NOT an official release.*